WHAT IS CLAIMED IS:

1. A method of controlling a processor that changes an execution sequence of instructions arranged in a program, the method comprising the steps of:

executing a second instruction that is placed after a first instruction in the program, prior to execution of the first instruction; and

when an address of first data to be executed by the first instruction is included in an address region of second data to be processed by the second instruction, overwriting an execution result of the first instruction on data corresponding to the address of the first data.

2. The method as claimed in claim 1, wherein:

the first instruction is a store instruction to store the first data into a storage unit; and

the second instruction is a load instruction to read out the second data from the storage unit.

3. The method as claimed in claim 1, wherein:

the step of executing the second

instruction includes the step of storing information for specifying a storage unit that stores an address of the second data to be processed by the second instruction and a result obtained by the execution of the second instruction; and

the step of overwriting is carried out in accordance with the address of the data to be processed by the second instruction and the information for specifying the storage unit.

4. The method as claimed in claim 3, further comprising the step of executing a third instruction so as to erase the address and the information for specifying the storage unit.

5. The method as claimed in claim 3, further comprising the step of executing a third instruction so as to erase either the address of the data to be processed by the second instruction or the information for specifying the storage unit.

6. The method as claimed in claim 1, wherein:

the step of executing the second instruction includes the step of storing identification information of a context to be processed by the second instruction; and

the step of overwriting is carried out in

accordance with the identification information of the context.

7. The method as claimed in claim 1, wherein the step of overwriting is carried out in accordance with an interrupt operation program, when the address of the first data to be processed by the first instruction is included in the address region of the second data to be processed by the second instruction.

8. The method as claimed in claim 1, wherein the step of overwriting is carried out in accordance with a program at a branch destination designated by executing a branch instruction, when the address of the first data to be processed by the first instruction is included in the address region of the second data to be processed by the second instruction.

9. A processor that executes programmed instructions, comprising:

a storage destination memory unit that stores a storage designation of a result obtained by executing a second instruction prior to execution of a first instruction, the second instruction being placed after the first instruction in a program;

a judgment unit that determines whether or

not an address of first data to be processed by the first instruction is included in an address region of second data to be processed by the second instruction; and

a data restoration unit that overwrites a result obtained by executing the first instruction on the second data corresponding to the address of the first data at the storage destination stored in the storage destination memory unit, when the judgment unit determines that the address of the first data is included in the address region of the second data.

10. The processor as claimed in claim 9, further comprising a storage unit that stores data,

wherein:

the first instruction is a store instruction to store the first data into the storage unit; and

the second instruction is a load instruction to read out the second data from the storage unit.

11. The processor as claimed in claim 9, further comprising a plurality of storage units,

the storage destination memory unit stores the information for specifying one of the storage units in which an address of the second data and the result obtained by executing the second instruction is stored.

12. The processor ass claimed in claim 9, further comprising a context information storage information for specifying a context to be processed by the second instruction,

wherein the judgment unit is activated, only when a context to be processed by the first instruction is determined to coincide with the context to be processed by the second instruction, in accordance with the information stored in the context information storage unit.

13. The processor as claimed in claim 9, wherein the data restoration unit performs an overwrite operation in accordance with an interrupt operation program, when the judgment unit determines that the address of the first data is included in the address region of the second data.

14. The processor as claimed in claim 9, wherein the data restoration unit performs an overwrite operation in accordance with a program at a branch destination designated through execution of a branch instruction, when the judgment unit determines that the address of the first data is included in the address region of the second data.

15. The processor as claimed in claim 9, further comprising a storage destination erase unit

that executes a third instruction so as to erase the storage destination stored in the storage destination memory unit.

16. The processor as claimed in claim 9, further comprising a storage destination erase unit that executes a third instruction so as to erase the storage destination stored in the storage destination memory unit.

17. A method of controlling a processor that controls execution of programmed instructions arranged in a program, the method comprising the steps of:

executing an instruction prior to execution of a branch instruction, the instruction being placed after the branch instruction in the program;

retaining an exception operation when necessity of the exception operation is detected in the step of executing;

performing the exception operation when the retained exception operation is needed in execution of an instruction at a branch destination selected through the execution of the branch instruction; and

returning to the program so as to continue the execution of the instruction at the branch destination when the exception operation is finished.

18. A method of controlling a processor that controls execution of instructions arranged in a program,

the method comprising the steps of:

executing an instruction prior to execution of a branch instruction, the instruction being placed after the branch instruction in the program;

retaining an exception operation when an exception start instruction that requires the exception operation is detected in the step of executing;

performing the exception operation when the retained exception operation is required in execution of an instruction at a branch destination selected through the execution of the branch instruction; and

returning to the program so as to sequentially execute the instructions starting from the exception start instruction, when the exception operation is finished.

19. The method as claimed in claim 17, wherein the step of performing the exception operation is carried out by executing an interrupt operation program.

20. The method as claimed in claim 17, wherein:

the step of retaining the exception operation includes the step of storing information

for performing the retained exception operation; and

the step of performing the exception operation is carried out in accordance with the stored information.

21. The method as claimed in claim 17, wherein:

the step of retaining the exception operation includes the step of allocating identification information to each set of data obtained as a result of a predetermined operation, the identification information indicating whether or not the corresponding set of data requires the exception operation; and

the step of performing the exception operation is carried out when a set of data that is determined to require the exception operation from the identification information is processed in the execution of the instruction at the branch destination selected through the execution of the branch instruction.

22. The method as claimed in claim 20, further comprising the step of executing a predetermined instruction so as to nullify the information for performing the retained exception operation.

23. The method as claimed in claim 21, further comprising the step of executing a predetermined instruction so as to nullify the identification information.

24. The method as claimed in claim 21, further comprising the step of executing a predetermined instruction so as to read out the identification information or to rewrite the identification information.

25. The method as claimed in claim 18, wherein the step of performing the exception operation is carried out by executing an interrupt operation program.

26. The method as claimed in claim 18, wherein:

the step of retaining the exception operation includes the step of storing information for performing the retained exception operation; and

the step of performing the exception operation is carried out in accordance with the stored information.

27. The method as claimed in claim 18, wherein:

the step of retaining the exception operation includes the step of allocating identification information to each set of data obtained as a result of a predetermined operation, the identification information indicating whether or not the corresponding set of data requires the exception operation; and

the step of performing the exception operation is carried out when a set of data that is determined to require the exception operation from the identification information is processed in the execution of the instruction at the branch destination selected through the execution of the branch instruction.

28. The method as claimed in claim 26, further comprising the step of executing a predetermined instruction so as to nullify the information for performing the retained exception operation.

29. The method as claimed in claim 27, further comprising the step of executing a predetermined instruction so as to nullify the identification information.

30. The method as claimed in claim 27, further comprising the step of executing a predetermined instruction so as to read out the identification information or to rewrite the identification information.

31. A processor that executes instructions arranged in a program, the processor comprising:

a control unit that controls an execution sequence so that an instruction placed after a branch instruction in the program is executed prior to execution of the branch instruction;

an exception inhibiting unit that retains an exception operation when necessity of the exception operation is detected during the execution of the instruction placed after the branch instruction;

an exception operation unit that performs the exception operation when the exception operation retained by the exception inhibiting unit is needed in execution of an instruction at a branch destination selected through execution of the branch instruction; and

a return unit that returns to the program when the exception operation is finished, and continues the execution of the instruction at the branch destination.

32. A processor that executes instructions arranged in a program, the processor

comprising:

a control unit that controls an execution sequence so that an instruction placed after a branch instruction in the program is executed prior to execution of the branch instruction;

an exception inhibiting unit that retains an exception operation when an exception start instruction that requires the exception operation is detected during the execution of the instruction placed after the branch instruction;

an exception operation unit that performs the exception operation when the exception operation retained by the exception inhibiting unit is needed in execution of an instruction at a branch destination selected through execution of the branch instruction; and

a return unit that returns to the program when the exception operation is finished, and sequentially executes the instructions starting from the exception start instruction.

33. The processor as claimed in claim 31, wherein the exception operation unit executes an interrupt operation program so as to perform the exception operation.

34. The processor as claimed in claim 31, further comprising a storage unit that stores information for performing the exception operation retained by the exception inhibiting unit,

wherein the exception operation unit

performs the exception operation in accordance with the information stored in the storage unit.

35. The processor as claimed in claim 31, wherein:

the exception inhibiting unit allocates identification information to each set of data obtained as a result of a predetermined operation, the identification information indicating whether or not the exception operation is required; and

the exception operation unit performs the exception operation, when data determined to require the exception operation in accordance with the identification information is processed in the execution of the instruction at the branch destination selected through the execution of the branch instruction.

36. The processor as claimed in claim 34, further comprising a history nullifying that executes a predetermined instruction so as to nullify the information for executing the retained exception operation.

37. The processor as claimed in claim 35, further comprising an identification information nullifying unit that executes a predetermined instruction so as to nullify the identification

information.

38. The processor as claimed in claim 35, further comprising:

an identification information read unit that executes a predetermined instruction so as to read out the identification information; and

an identification information rewrite unit that executes a predetermined instruction so as to rewrite the identification information.

39. The processor as claimed in claim 32, wherein the exception operation unit executes an interrupt operation program so as to perform the exception operation.

40. The processor as claimed in claim 32, further comprising a storage unit that stores information for performing the exception operation retained by the exception inhibiting unit,

wherein the exception operation unit performs the exception operation in accordance with the information stored in the storage unit.

41. The processor as claimed in claim 32,

wherein:

the exception inhibiting unit allocates identification information to each set of data obtained as a result of a predetermined operation, the identification information indicating whether or not the exception operation is required; and

the exception operation unit performs the exception operation, when data determined to require the exception operation in accordance with the identification information is processed in the execution of the instruction at the branch destination selected through the execution of the branch instruction.

42. The processor as claimed in claim 40, further comprising a history nullifying that executes a predetermined instruction so as to nullify the information for executing the retained exception operation.

43. The processor as claimed in claim 41, further comprising an identification information nullifying unit that executes a predetermined instruction so as to nullify the identification information.

44. The processor as claimed in claim 41, further comprising:

an identification information read unit that executes a predetermined instruction so as to read out the identification information; and

an identification information rewrite unit that executes a predetermined instruction so as to rewrite the identification information.

45. A method of controlling execution of instructions in a program, the method comprising the steps of:

executing an instruction prior to execution of a branch instruction, the instruction being placed after the branch instruction in the program;

retaining a break operation when necessity to suspend execution of the program is detected in the step of executing the instruction; and

performing the break operation when the retained break operation is required in execution of an instruction at a branch destination selected through the execution of the branch instruction.

46. The method as claimed in claim 45, wherein:

the step of retaining a break operation includes the step of storing information for performing the retained break operation; and

the step of performing the break operation is carried out in accordance with the stored information.

47. The method as claimed in claim 46, further comprising the step of nullifying the stored information.

48. The method as claimed in claim 45, wherein:

the step of retaining a break operation includes the step of setting a predetermined value into a flag; and

the step of performing the break operation includes the step of referring to the value of the flag so as to determine whether or not the retained break operation is needed in execution of an instruction at a branch destination selected through the execution of the branch instruction.

49. The method as claimed in claim 48, further comprising the step of executing a predetermined instruction so as to nullify the value of the flag.

50. The method as claimed in claim 45, wherein the step of performing the break operation includes the step of executing the instruction at the branch instruction selected through the execution of the branch instruction, in accordance with an interrupt operation program.

51. A processor that executes instructions in a program, the processor comprising:

an exception inhibiting unit that retains a break operation when necessity of suspending execution of the program is detected in execution of a predetermined instruction prior to execution of a branch instruction, the predetermined instruction being placed after the branch instruction in the program; and

a break operation unit that performs the break operation when the break operation retained by the exception inhibiting unit is required in execution of an instruction at a branch destination selected through the execution of the branch instruction.

52. The processor as claimed in claim 51, further comprising a storage unit that stores information for performing the retained break operation,

wherein the break operation unit performs the break operation in accordance with the information stored in the storage unit.

53. The processor as claimed in claim 52, further comprising a nullifying unit that nullifies the information stored in the storage unit.

54. The processor as claimed in claim 51, further comprising a flag,

wherein:

the exception inhibiting unit sets a predetermined value in the flag; and

the break operation unit refers to the value of the flag so as to determine whether or not the retained break operation is needed in the execution of the instruction at the branch destination selected through the execution of the branch instruction.

55. The processor as claimed in claim 54, further comprising a flag nullifying unit that executes a predetermined instruction so as to nullify the flag.

56. The processor as claimed in claim 51, further comprising an interrupt operation unit that executes the instruction at the branch destination selected through the execution of the branch instruction, in accordance with an interrupt operation program.